AFRL-RI-RS-TR-2009-54
Final Technical Report
February 2009

ENTITY BASES: LARGE-SCALE KNOWLEDGEBASES FOR INTELLIGENCE DATA

University of Southern California

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

NOTICE AND SIGNATURE PAGE

AFRL-RI-RS-TR-2009-54 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/s/	/s/
ROGER J. DZIEGIEL, Jr.	JOSEPH CAMERA, Chief
Work Unit Manager	Information & Intelligence Exploitation Division
	Information Directorate

REPORT DOCUMENTATION PAGE	*Form Approved* **OMB No. 0704-0188**

1. REPORT DATE *(DD-MM-YYYY)* FEB 09	**2. REPORT TYPE** Final	**3. DATES COVERED** *(From - To)* May 05 – Dec 08

4. TITLE AND SUBTITLE ENTITY BASES: LARGE-SCALE KNOWLEDGEBASES FOR INTELLIGENCE DATA	**5a. CONTRACT NUMBER** FA8750-05-C-0116
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER 61101E
6. AUTHOR(S) Craig Knoblock, Steven Minton and Ching-Chien Chen	**5d. PROJECT NUMBER** RAPD
	5e. TASK NUMBER 02
	5f. WORK UNIT NUMBER 01
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) PRIMARY: University of Southern California, 4676 Admiralty Way, Marina del Rey CA 90292-6695; SUB: Fetch Technologies & Geosemble Tech, 841 Apollo St., El Segunda CA 90245	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) AFRL/RIED, 525 Brooks Rd., Rome NY 13441-4505	**10. SPONSOR/MONITOR'S ACRONYM(S)**
	11. SPONSORING/MONITORING AGENCY REPORT NUMBER AFRL-RI-RS-TR-2009-54

12. DISTRIBUTION AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED. PA# 09-0684

13. SUPPLEMENTARY NOTES

14. ABSTRACT
This report describes new technology for rapidly integrating information from heterogeneous sources. First, we consider the problem of integrating records about entities harvested from multiple sources. We address this problem by developing the technology to build massive entity knowledgebases, which we call EntityBases. The key idea is to create a comprehensive knowledgebase for the entities of interest. In order to build such a knowledge base, we address the issues of linking entities with noisy, multi-valued attributes obtained from heterogeneous sources and providing a virtual repository that can be efficiently queried. This report describes how we have addressed these issues and shows how an EntityBase can be used for understanding and linking text documents. We also consider the problem of on-demand information integration and describe a novel smart copy and paste (SCP) architecture that seamlessly combines the design-time and run-time aspects of data integration.

15. SUBJECT TERMS
Entity Resolution, entity consolidation, information integration, machine learning, artificial intelligence

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Roger J. Dziegiel, Jr.
a. REPORT U	**b. ABSTRACT** U	**c. THIS PAGE** U	UU	28	**19b. TELEPHONE NUMBER (*Include area code*)** N/A

Table of Contents

List of Figures

1 Summary

Recent advances in networking technology, especially the Internet, have made a huge amount of data available about entities, such as people, places and organizations. Even so, our ability to use the vast quantities of data on-line for identifying the references in text documents or linking information across sources remains primitive. Finding entities of interest in real-time is challenging, due to the difficulty of integrating and querying multiple databases, web sites, and document repositories.

In this project, we developed technology to rapidly aggregate and organize data from multiple sources. In particular we focused on two specific types of technology. First, we developed an approach that makes it possible to rapidly create large-scale, well-organized, entity knowledgebases -- which we refer to as EntityBases. EntityBases enable information to be integrated on a scale that far exceeds current capabilities. The resulting EntityBases can then be used for a variety of applications (e.g., document understanding or data mining). The EntityBases prototype described in this report represents a novel approach to integrating information from numerous heterogeneoussources.

Second, also developed technology suitable for emergency response or ad hoc collaboration, where it is critical to reduce the overhead in *integrating* data. Here, the goal is often to rapidly integrate ``enough" data to answer a specific question. Ideally, one could perform the entire process *interactively* under one unified interface: defining extractors and wrappers for sources, creating a mediated schema, and adding schema mappings --- while seeing how these impact the integrated view of the data, and refining the design accordingly. To address this goal, we worked with Zack Ives of the University of Pennsylvana to develop a novel smart copy and paste (SCP) model and architecture for seamlessly combining the design-time and run-time aspects of data integration. In this report we describe an initial prototype, the CopyCat system. In CopyCat, the user does not need special tools for the different stages of integration: instead, the system watches as the user copies data from applications (including the Web browser) and pastes them into CopyCat 's spreadsheet-like workspace. CopyCat generalizes these actions and presents proposed auto-completions, each with an explanation in the form of provenance. The user provides feedback on these suggestions --- through either direct interactions or further copy-and-paste operations --- and the system learns from this feedback. This report provides an overview of our prototype system, and identifies key research challenges in achieving SCP in its full generality.

2 Introduction

2.1 EntityBases Overview

There is a great deal of information available about entities -- people, places and things – from multiple data sources. These advances offer the possibility of creating much more intelligent decision support and situation awareness applications. However, aggregating information about entities from multiple sources, such as databases, web sites, or document repositories, raises challenging technical problems.

These technical problems are central to this project, which focuses on using multiple data sources to make sense of a situation. When data is distributed in multiple heterogeneous sources, each of which uses different data formats and/or terminology, it is difficult to aggregate the data and unambiguously interpret it. Our Entitybase architecture represents a novel approach to collecting and integrating information from numerous heterogeneous sources. An EntityBase consolidates data, so that references to the same entity in multiple information sources can be resolved. The consolidation process can resolve references in different formats (e.g. "Joe Smith" vs. "Smith, J.E.", or "IBM" vs. "International Business Machines Corp."), represent uncertainty, accommodate aliases, and support continuous updates (so information is not lost when two consolidated references are later determined to refer to two distinct entities).

2.1.1 Representing EntityBases

There are several fundamental representation issues that must be addressed in building an EntityBase. First, since an EntityBase is constructed from multiple heterogeneous sources, we must be able to support multi-valued attributes for describing an entity. Second, different sources will describe these attributes at different levels of detail. Hence, an EntityBase must also be able to support the representation of the attributes at different levels of granularity. For example, one source may represent an address as a single unit, while another source may break it down into the street address, city, country, and postal code.

2.1.2 Organizing EntityBases

The capability to consolidate multiple references to the same individual entity collected from different information sources is a central aspect of the EntityBase architecture. Previous research has developed a foundation for statistically linking references across multiple databases, referred to variously as record linkage, consolidation or object identification (Fellegi and Sunter 1969; Winkler 1994; Tejada et al. 2001; Bilenko and Mooney 2003). The challenge here is to build this technology into a practical architecture for large-scale information repositories. We have developed an integrated EntityBase system that supports the statistical consolidation process "invisibly" as an EntityBase is populated, enables users to easily understand and analyze results, enables queries to be executed efficiently, and is robust to updates so that references can be both consolidated as new information becomes available.

2.1.3 Querying EntityBases

In previous research, we and others (Duschka 1997; Ambite et al. 2001; Halevy 2001) have addressed many of the theoretical problems underlying virtual databases (i.e. mediator systems that integrate distributed, heterogeneous sources). Nevertheless, building large-scale virtual databases remains challenging in practice because it is difficult to model complex data relationships and potentially expensive to execute arbitrary queries against virtual databases. Our goal is to address these specific problems. By focusing only on entities, our architecture simplifies the modeling issues and improves the tractability of query processing.

2.1.4 Exploiting Geospatial Context

Most entities in the world are associated with some geospatial location. This location could be a point or even a region and we refer to the associated location as the geospatial extent of an entity. We automatically determine the geospatial extent of an entity and then use this extent as an additional source of information for linking new sources of information into the system. For example, if a new record is added to the system and the area code of the phone number indicates the record is in a particular region, then it would be less likely to match against an entity located in a different region.

2.2 Smart Copy and Paste Overview

Today's data integration tools require substantial up-front investment at design-time — understanding source schemas, creating a mediated schema, defining schema mappings — before a runtime system can be used to produce results or handle updates. Thus integrating data is a long and laborious process: by the time good results are achieved, the "window of opportunity" where the data is most useful may have elapsed, or application requirements might have changed! This has led to a series of research efforts designed to reduce the initial design-time effort (perhaps sacrificing some result quality): dataspaces (Franklin et al., 2005); "pay as you go" data integration (Sarma et al., 2008); "best effort" integration and extraction (McCann et al., 2008, Shen et al., 2008); and peer-to-peer query answering with composable (Halevy et al., 2003, Ives et al., 2008) or probabilistic (Dong et al., 2007) schema mappings. Such work shares two tenets: (1) provide basic functionality even when little user effort has been invested, and more functionality as more human input is given; (2) leverage and reuse human effort where possible. The expectation here is that the integration process will be done by iteratively switching between design-time and runtime, until sufficient result quality is achieved.

We believe that, for many "best effort" integration applications where time is of the essence, an even better approach is to combine design-time and runtime aspects into a single interactive process. We should be able to add sources, design a target schema or even a one-off query, specify mappings or other operators, see results, and refine those results or the schema—all on-the-fly, with a single seamless mode of interaction. This enables a data integrator to develop an understanding of the data sources as he or she is integrating them; and to assemble and revise the integrated or mediated schema and the mappings in accordance with this understanding.

Importantly, the integrator can directly see the impact of design choices on the integrated data (which is also "explained" by visualizing its provenance (Buneman et al., 1997, Cui, 2001, Ives et al., 2008)) as they work.

To support this type of integration, we propose and implement a scheme we term smart copy and paste (SCP): the integrator follows the familiar model of copying items of interest from existing applications (Web browser, office applications, etc.), and pasting them into a dynamic, spreadsheet-like "workspace." As the integrator pastes content, the system attempts to infer potential generalizations of the user actions, and it shows suggestions along the lines of an "auto-complete" in Microsoft Word. These are intuitively the results from proposed information extractors (wrappers) over data sources, and from potential mappings (transformations expressed as queries or constraints) across sources. 1 The integrator provides feedback to the system by accepting the auto-complete suggestions (or portions thereof), or by simply ignoring the suggestions and pasting further content. Finally, an SCP system should include built-in interfaces to data visualization tools such as Google Maps, as well as the ability to export data to standard formats. To illustrate how SCP works, we sketch a basic usage scenario below for our prototype system, CopyCat (Copy and conCatenate).

EXAMPLE 1. Consider a hurricane relief scenario: FEMA needs to establish connections to supplies, shelters, road conditions, damage regions, etc., and begin making decisions. This is a best effort data integration problem: it is more critical to immediately get data (even with a few errors) than it is to get perfect results.

Suppose one integration task is to take a list of shelters from a television news Web site; combine it with the shelters' contact information from a spreadsheet; and plot the shelters on a map. In CopyCat, a data integrator would load the page of shelters into her Web browser. She would select and copy the first item, then paste it into the CopyCat workspace. The system would try to generalize the integrator's action by extracting other shelters from the same page and proposing new rows on the workspace. She might accept these new rows and then copy the first shelter's name into Google Maps to get its full address and geocode. She would paste the resulting information into the workspace, in the same row as the first shelter. The system would again generalize, taking all subsequent shelter names, feeding them to the map site, and retrieving the matching addresses and geocodes. In some cases the shelter name may be ambiguous and might return multiple answers: here CopyCat would show the alternatives and allow the integrator to select the appropriate location.

Finally, the integrator would load the spreadsheet of contacts, and copy the contact info that best matches the first shelter (e.g., approximately matching the name and address), then paste it into the workspace. Here the match might not be a direct lookup, but rather the result of approximate record linking techniques, which determine the contact that best matches each shelter. CopyCat learns the best combination of heuristics for this case of record linking, via a combination of generalizing examples (the integrator might paste matches for several shelters) and accepting feedback (she might accept or reject suggested matches).

Ultimately, a complete table would be assembled in the workspace. This could be made persistent saved as an integrated view of the data, enabling user or application queries over a unified representation. In our example, the data would also be exported to a Google Maps visualization.

Note that interaction with an SCP system is quite different from running the usual series of schema matching, mapping creation, and query processing tools: here, the means of specifying mappings is to use existing applications, the clipboard, and the SCP workspace; the system's semi-automatic features suggest auto-completions that can be ignored at no cost, or accepted or refined if beneficial. The integrator gets to see and provide feedback on data the moment she provides input to the system. We describe the user interface in detail later in this report.

2.3 RoadMap

In the following sections of this report we begin by describing the EntityBase architecture. We include a motivating example, describe our approach to representing, organizing, and querying an EntityBase, and present our approach to exploiting the geospatial context. This work was carried out jointly by USC ISI, Fetch Technologies, and Geosemble Technologies.

We then describe our design and implementation of our CopyCat prototype, our experiences with it, and a discussion of its significance. This work was carried out jointly by USC ISI, Fetch Technologies, and, in a separately funded effort, the University of Pennsylvania.

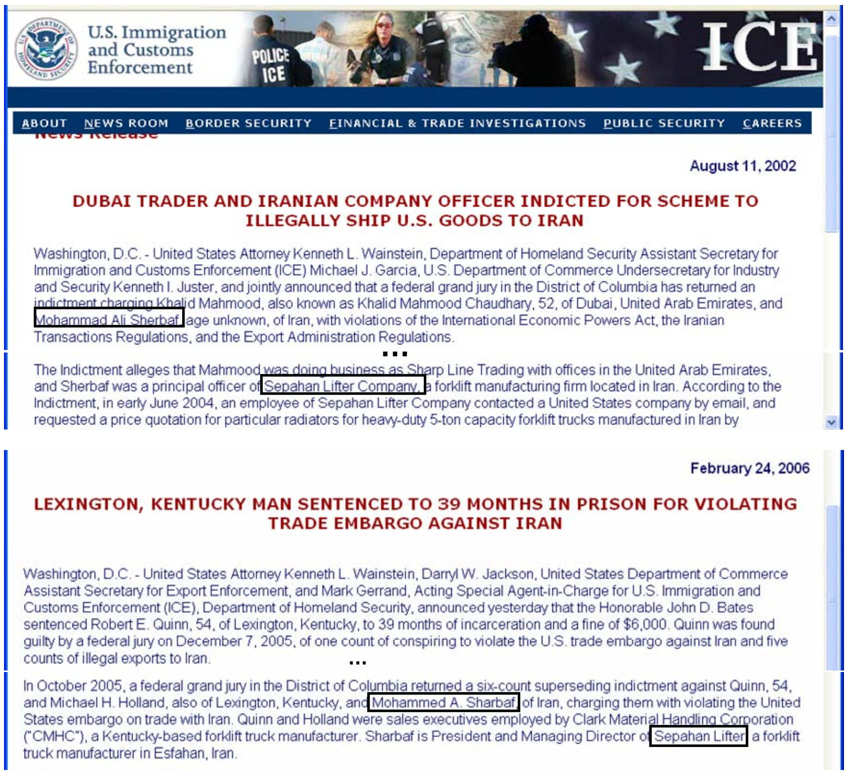
3 EntityBases: Technical Description

3.1 Motivating Example

Consider the following real-world example of the need for EntityBases. Figure 1 shows extracts of two news releases from the U.S. Immigration and Customs Enforcement, an agency of the U.S Department of Homeland Security. The documents describe a case involving several individuals and companies accused of illegal exports to Iran.

Entity extraction software, like Inxight's SmartDiscovery system, can automatically extract entities, such as company, person or location names, from these documents. For example, "Mohammad Ali Sherbaf", "Kenneth L. Wainstein", and "Khalid Mahmood Chaudhary", etc., would be recognized as person names. Similarly, "Sepahan Lifter Company", "Sharp Line Trading", and "Clark Material Handling Corporation" would be labeled as companies, and "Esfahan" as a city and "Iran" as a country.

However, simple entity extraction is not enough. The relationship between these two documents cannot be established without the kind of record linkage reasoning that our EntityBase provides. In particular, note that the 2002 document refers to one of the key persons involved in the case as "Mohammad Ali Sherbaf" while the 2006 document as "Mohammad A. Sharbaf" (even though the documents come from the same government agency). Different transliterations of foreign names, such as in this case, would foil simple match techniques. The multiplicity of names that refer to the same real-world entity is not limited to people --- other entities, such as companies and locations, exhibit the same phenomenon. For example, both "Isfahan" and "Esfahan" are common transliterations for the same Iranian city.


U.S. Immigration and Customs Enforcement
POLICE ICE
ICE
ABOUT
NEWS ROOM
BORDER SECURITY
FINANCIAL & TRADE INVESTIGATIONS
PUBLIC SECURITY
CAREERS

August 11, 2002

DUBAI TRADER AND IRANIAN COMPANY OFFICER INDICTED FOR SCHEME TO ILLEGALLY SHIP U.S. GOODS TO IRAN

Washington, D.C. - United States Attorney Kenneth L. Wainstein, Department of Homeland Security Assistant Secretary for Immigration and Customs Enforcement (ICE) Michael J. Garcia, U.S. Department of Commerce Undersecretary for Industry and Security Kenneth I. Juster, and jointly announced that a federal grand jury in the District of Columbia has returned an indictment charging Khalid Mahmood, also known as Khalid Mahmood Chaudhary, 52, of Dubai, United Arab Emirates, and $\boxed{\text{Mohammad Ali Sherbaf}}$ age unknown, of Iran, with violations of the International Economic Powers Act, the Iranian Transactions Regulations, and the Export Administration Regulations.

...

The Indictment alleges that Mahmood was doing business as Sharp Line Trading with offices in the United Arab Emirates, and Sherbaf was a principal officer of $\boxed{\text{Sepahan Lifter Company,}}$ a forklift manufacturing firm located in Iran. According to the Indictment, in early June 2004, an employee of Sepahan Lifter Company contacted a United States company by email, and requested a price quotation for particular radiators for heavy-duty 5-ton capacity forklift trucks manufactured in Iran by

February 24, 2006

LEXINGTON, KENTUCKY MAN SENTENCED TO 39 MONTHS IN PRISON FOR VIOLATING TRADE EMBARGO AGAINST IRAN

Washington, D.C. - United States Attorney Kenneth L. Wainstein, Darryl W. Jackson, United States Department of Commerce Assistant Secretary for Export Enforcement, and Mark Gerrand, Acting Special Agent-in-Charge for U.S. Immigration and Customs Enforcement (ICE), Department of Homeland Security, announced yesterday that the Honorable John D. Bates sentenced Robert E. Quinn, 54, of Lexington, Kentucky, to 39 months of incarceration and a fine of $6,000. Quinn was found guilty by a federal jury on December 7, 2005, of one count of conspiring to violate the U.S. trade embargo against Iran and five counts of illegal exports to Iran.

...

In October 2005, a federal grand jury in the District of Columbia returned a six-count superseding indictment against Quinn, 54, and Michael H. Holland, also of Lexington, Kentucky, and $\boxed{\text{Mohammed A. Sharbaf}}$ of Iran, charging them with violating the United States embargo on trade with Iran. Quinn and Holland were sales executives employed by Clark Material Handling Corporation ("CMHC"), a Kentucky-based forklift truck manufacturer. Sharbaf is President and Managing Director of $\boxed{\text{Sepahan Lifter}}$ a forklift truck manufacturer in Esfahan, Iran.

Figure 1 Different documents with the same real-world entities

The EntityBase approach uses previously gathered knowledge to help differentiate the entities that appear in documents like these and provide additional information. Our EntityBase can recognize that "Mohammad Ali Sherbaf" and "Mohammad A. Sharbaf" are the same person and that "Sepahan Lifter Company", "Sepahan Lifter", "Sepahan Lifter Co." refer to the same company. Moreover, the EntityBase also shows that this company has its headquarters in "Nos. 27 and 29, Malekian Alley, North Iranshahr Ave., Tehran (15847)" and its factory in "Mahyaran Industrial Town, Isfahan"; that its commercial manager is "Mohammad Kharazi" and its headquarters' phone and fax numbers are (+98-21) 8830360-1 and (+98-21) 8839643, respectively. At the same time, the EntityBase shows that "Sepahan Lifter Company", "Behsazan Granite Sepahan Co.", and "Rahgostar Nakhostin Sepahan Co." are different companies that are all located in Isfahan, Iran.

3.2 Represention in EntityBases

In order to decide how to represent entities, two fundamental issues must be addressed. The first is the multivalued nature of entity information. For example, a company may have multiple phone numbers or multiple addresses. This complicates the issue of "record" linkage, because the data is no longer a "record" (i.e., a tuple or row in a database), but an object with multi-valued attributes. The second issue is the level of schema granularity. Normalizing information into finer levels of granularity – while seemingly more precise – is not always possible and can potentially result in a loss of information. In the following subsections, we describe our solutions to these issues.

3.2.1 Multi-valued attributes

In practice, many entity attributes are multi-valued. For example, most businesses have multiple phone numbers. Many people are known by multiple names (e.g., maiden name and married name). Many publications have multiple authors. This feature of real world entities requires a more general representation than the traditional records. As a more detailed example, consider Figure 2 that shows some of the multi-valued attributes of our running example.

Company name	Key person
Sepahan Lifter Company	Mohammad A. Sharbaf
Sepahan Lifter Co.	Mohammad Ali Sherbaf
Sepahan	M.A. Sherbaf

Figure 2: Entity with multi-valued attributes

We represent an entity as a set of multi-valued attributes. For example, we represent the above information as {[NAME: {"Sepahan Lifter Company", "Sepahan Lifter Co.", "Sepahan"}], [KEYPERSON: {"Mohammad Ali Sherbaf", "Mohammad A. Sharbaf", … }], … }. This representation reduces the amount of data to be stored to only the unique attribute values, but requires more sophisticated matching techniques. We need to move from "record" linkage to "entity" linkage.

3.2.2 Level of schema granularity

Another representational issue is the level of schema granularity, an issue that arises due to the heterogeneous origins of the data and the inability to precisely parse all types of real-world information. Data in the EntityBase comes from different sources which have different schemas. For example, one source may break down address into street, city, and state while another may just have all of this data in one attribute (e.g., address). Because of this, we must decide at what level of granularity the EntityBase will normalize the information. Generally speaking, there are two possible options: fine-grain or coarse-grain.

The fine-grain option, where one might capture attributes such as street, city, state, suite number, and zip provides more information about a match (i.e., identifies the specific attribute a match occurs on), but assumes that all information can be neatly deconstructed, or that it is possible to store ambiguous information when information cannot be reliably parsed. For example, consider the sequence of tokens “Mohammad Ali Sherbaf”. The fine-grain option assumes that we can parse this name, in particular that we know the first name is “Mohammad” and that the last name is either “Ali Sherbaf” or that the middle name is “Ali” and the last name is “Sherbaf”.

The coarse grain option, on the other hand, eliminates the need to unambiguously parse the data. If we simply treat the sequence of tokens as just that – a sequence of tokens (i.e., a document) – we have no need to resolve ambiguous parses when storing the data. However, this does mean that we must parse the data at run-time (i.e., query time), somewhat troubling from a performance standpoint.

In EntityBases, we chose to use a hybrid approach that exploits both the coarse and fine-grained representation. The coarse-grained representation is used during the initial phase of generating candidate matches since this initial matching is based on token overlap. And then both the coarse-grained and fine-grained representations are available for reasoning in the detailed matching process. Blocking is designed to be efficient, relying on simpler (e.g., token-based) metrics in order to identify a concise, quality candidate set. In contrast, linkage is designed to focus on accuracy, performing a more careful analysis of each candidate, including evaluation of the parsed data.

3.3 Organizing EntityBases

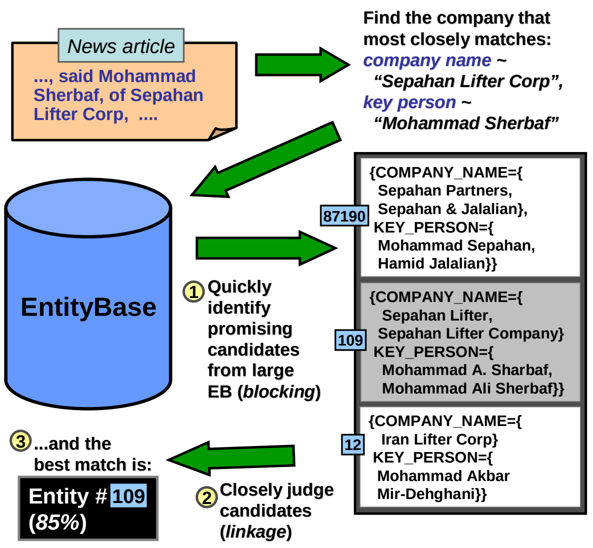
An EntityBase provides two main capabilities: (1) entity matching, that is, the ability to match the relevant entities given a query, and (2) entity creation/update, that is, the ability to decide whether newly acquired information belongs to an existing entity or constitutes a new entity. Note that entity creation or update requires matching as part of its process. We now present an overview of both the match and update processes.

3.3.1 Matching

There are two major phases of entity matching: blocking and linkage. The purpose of the blocking phase (Michelson and Knoblock 2006) is to very quickly identify the most promising candidates from a much larger set of possible candidates. In our system, blocking relies on simple, yet efficient techniques for reducing the space of possible candidates, for example by using token-based distance metrics (Jaccard coefficient, TF-IDF, etc). The purpose of the linkage phase is to do a more detailed evaluation of the incoming record/query to the candidate entity. This is accomplished through a variety of more sophisticated transformations (e.g., alignment of parsed representations of the data), which can be more accurate but require more computational resources.

Blocking and linkage complement each other, with the former focusing on performance and the latter focusing on quality. A key challenge is to ensure that the efficiency of blocking does not result in false negatives, as this would handicap the linkage phase. At the same time, blocking cannot produce too many false positives, as that would swamp the linkage phase with computational demands.

Figure 3 shows an example of the matching process. An incoming news document mentions the company “Sepahan Lifter Corp” as well as “Mohammad Sherbaf”. This information can be used to query the EntityBase (which consists of millions of company entities). The blocking process efficiently identifies candidates that appear consistent with the information we know. As the example shows, many of the candidates have tokens that also appear in the query, so even applying a Jaccard-style metric (i.e., token overlap) or TF-IDF would be sufficient to yield the candidates shown.



News article
..., said Mohammad Sherbaf, of Sepahan Lifter Corp, ...
Find the company that most closely matches:
company name ~ "Sepahan Lifter Corp",
key person ~ "Mohammad Sherbaf"
EntityBase
87190
{COMPANY_NAME={ Sepahan Partners, Sepahan & Jalalian}, KEY_PERSON={ Mohammad Sepahan, Hamid Jalalian}}
109
{COMPANY_NAME={ Sepahan Lifter, Sepahan Lifter Company} KEY_PERSON={ Mohammad A. Sharbaf, Mohammad Ali Sherbaf}}
12
{COMPANY_NAME={ Iran Lifter Corp} KEY_PERSON={ Mohammad Akbar Mir-Dehghani}}
1 Quickly identify promising candidates from large EB (blocking)
2 Closely judge candidates (linkage)
3 ...and the best match is:
Entity #109 (85%)

Figure 3: Entity blocking and linkage

The linkage process (Minton et al. 2005) then compares the data in greater detail, parsing the incoming query to realize that "Corp" is a previously unseen term associated with the company's name, and that "Ali" is missing from Mohammad Sherbaf's name. It also evaluates the other candidates and identifies similar differences. In evaluating the candidates, the linkage phase associates metric scores to quantify the similarity (or lack there of). A second part of the linkage process evaluates the similarities/dissimilarities and then judges the implications of such scores. For example, the linkage process could have identified that Corp is just a common company formation acronym (like "Inc." or "LLP") and that the missing "Ali" from the person's name is not critical (as opposed to a mismatch on last name, for example).

3.3.2 Updating

Recall that the EntityBase supports both queries and updates. Thus far, we have discussed the matching process, which plays a key role in both phases. However, during the update phase, we must also insert new data. This is not necessarily trivial, as new information may cause us to re-evaluate current entities. In particular, new data may cause two previously distinct entities to merge. Typically, the merging scenario arises when new information contains strong matches to two different entities. For example, in Figure 3 above, notice that entity #12 (Iran Lifter Corp) is different from entity #109 (Sepahan Lifter Company). However, suppose that we update the EntityBase with a new source of Iran company information and that one of those incoming records suggests that Mohammad Akbar Mir-Dehghani is a key person of Sepahan Lifter. The entity match phase would result in both entity #12 and entity #109 receiving high match confidences. At that point, the EntityBase could decide to merge those two entities together.

3.4 Querying EntityBases

Our EntityBase architecture provides access to the available information about entities from both local and remote sources. Even with the rapidly declining cost of storage, it is not possible to materialize all the entity information in a local warehouse due to policy, control, and security considerations. In addition data may be too volatile to store and must be accessed live when queried, such as the changing stock price of a company. Therefore, our EntityBase is organized as a virtual repository that integrates both locally materialized data and remote data.

Figure 4 shows the EntityBase query architecture. The Local Entity Repository (LER) materializes the identifying attributes of the entities in order to perform record linkage reasoning efficiently. The concept of an entity-identifying attribute is broader than the concept of a key in relational databases. For example, the name, address and phone number are useful entity-identifying attributes, but none of them are keys.

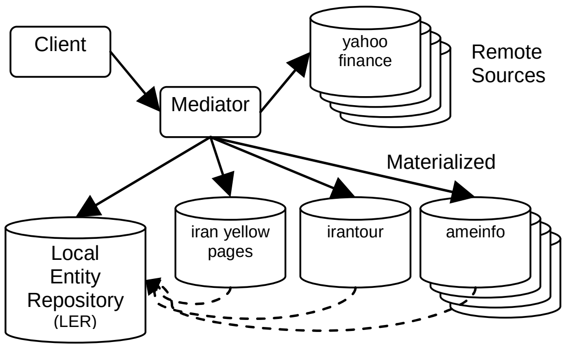


Figure 4: EntityBase data integration architecture

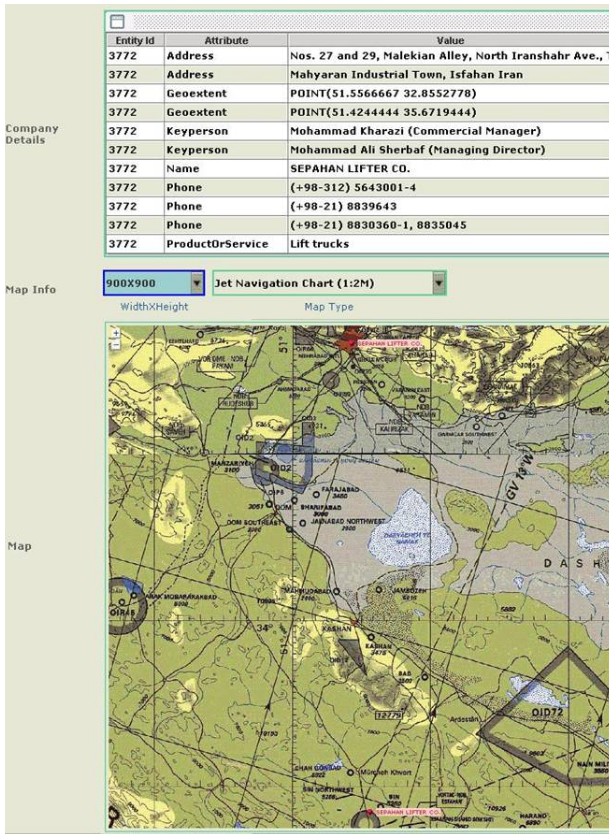
Additional information about entities can also be materialized, but it is not copied into LER for performance. For example, images and reports may be associated with entities, but they would not be useful for efficient record linkage. Finally, other information may reside in remote sources.

The EntityBase uses a mediator to orchestrate all these local and remote sources. The crucial idea in a mediator system is to use a mediated schema to assign common semantics to the data from the diverse sources. A human analyst, or a client program, queries the entity base using the mediated schema without worrying how the information is represented in the sources. We have built upon our previous work in the Prometheus mediator (Thakkar et al. 2005) to define our mediator for the Entity-Base. In our mediator the contents of sources are declaratively expressed as Datalog rules.

The EntityBase supports two query scenarios:

- Free-Form Querying: A human analyst or a client program can pose arbitrary queries over the mediated schema to the EntityBase.
- Document Matching: Some partial entity information is extracted from a document and additional information about the entity is required.

The EntityBase mediator handles both cases uniformly. First, the mediator invokes the entity matching module with the constraints appearing in the query. In free-form querying the constraints are the selections on entity-identifying attributes appearing in the query. In document matching, the partial entity information triggers entity matching. Next, the mediator retrieves the requested information from local or remote sources corresponding to the set of candidate entities produced by the entity matching module.



Company Details

Entity Id	Attribute	Value
3772	Address	Nos. 27 and 29, Malekian Alley, North Iranshahr Ave.,
3772	Address	Mahyaran Industrial Town, Isfahan Iran
3772	Geoextent	POINT(51.5566667 32.8552778)
3772	Geoextent	POINT(51.4244444 35.6719444)
3772	Keyperson	Mohammad Kharazi (Commercial Manager)
3772	Keyperson	Mohammad Ali Sherbaf (Managing Director)
3772	Name	SEPAHAN LIFTER CO.
3772	Phone	(+98-312) 5643001-4
3772	Phone	(+98-21) 8839643
3772	Phone	(+98-21) 8830360-1, 8835045
3772	ProductOrService	Lift trucks

Map Info: 900X900 (WidthXHeight), Jet Navigation Chart (1:2M) (Map Type)

Figure 5: EntityBase analyst query interface

Figure 5 shows the analyst query interface to the EntityBase. This graphical interface was developed using the Heracles system (Ambite et al. 2005). The Figure shows some detailed data about the Sepahan Lifter Company including geospatial locations. Note that the company has two addresses, one in Teheran and one in Isfahan, and thus the map shows these two locations.

3.5 Exploiting Geospatial Context in EntityBases

When querying and matching entities, we can exploit the geospatial extents of the entities to help identify and assess possible matches. Consider our running example shown in Figure 3, where a document understanding system queries the EntityBase to match the company "Sepahan Lifter Corp" and the person "Mohammad Sherbaf," which are extracted from a document. The EntityBase may provide many candidates (e.g., entity #12, #109 and #87190) even after applying the blocking process. However, by exploiting the geospatial extents of these entities, the system can deduce additional information to narrow down the relevant entities. For example, the company mentioned in the document is located within area X shown in Figure 6. The system infers that entity #109 is located within area X as well (based on its "phone" attribute). It also infers that the companies of the entities #12 and #87190 are located within area Y (based on their associated "phone" attribute). This in turn implies lower similarity between the company mentioned in the document and the two entities #12 and #87190.

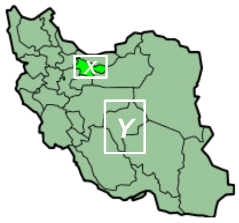


Figure 6: Geocoordinates of the companies in Iran

Unfortunately, identifying the exact geospatial extent of an entity is not straightforward. Essentially, we need to transform a record's textual geographic information (e.g., mailing addresses) to spatial extents (e.g., geocoordinates). A straightforward way to compute geocoordinates of a company is using a geocoder with the mailing address as input (Bakshi et al. 2004). Typically, a geocoder determines the geocoordinates of an address by utilizing a comprehensive spatial database (e.g., labeled road network data). However, such a comprehensive, well-formatted spatial database does not exist or is not accessible for many countries. Additionally, addresses are non-standard (e.g., "No. 1780, Opp.to The Main Gate of England Embassy Garden, Off the Dolat St., Shariati Ave., Tehran, Iran"), incomplete, and sometimes do not exist for a given record (e.g., only the phone number exists).

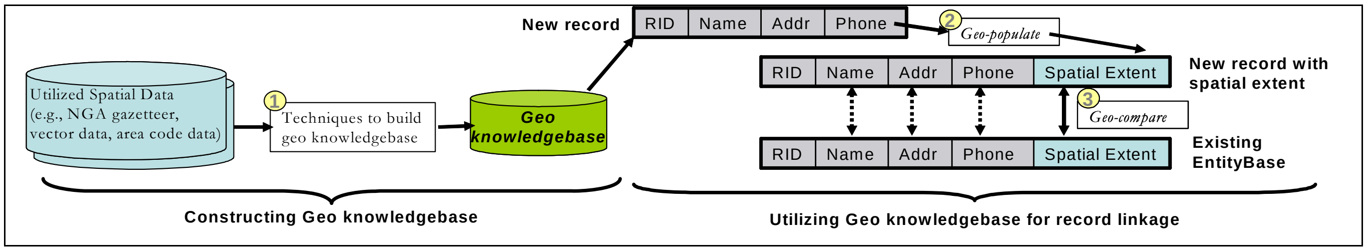
Toward this end, we utilize various techniques to build a geospatial knowledgebase of an area from available public data. The geospatial knowledgebase contains abundant (inferred) spatial datasets, such as landmarks, road network data, zip code maps, and area code maps. To illustrate, consider the scenario that the area code data for Iran is not available. We can use the technique proposed in (Sharifzadeh et al. 2003) to approximate the area code regions and store them in the geospatial knowledgebase. This technique utilizes classification techniques (such as Support Vector Machines) based on a set of training data to build approximate thematic maps (e.g., area code maps). For example, the training data can be cities with spatial coordinates and telephone area code attributes. Spatial classification of the training data (geocoordinates labeled with phone area code) produces an approximate thematic map of the phone area code regions.

There are many online public data sources, such as the NGA gazetteer database[1] that can provide the labeled training points. Again, consider building area code maps for the country of Iran as in our case study. We apply the technique for building thematic maps to three datasets we collected for Iran: (1) Iran area codes and corresponding cities, which are available from IranAtom[2], (2) NGA gazetteer database that provides the coordinates of populated points (including cities) around the world, and (3) Iran province information[3] that provides the spatial bounding box for every province in Iran. Finally, we store the approximate area code vector maps into the geospatial knowledgebase in Oracle 10g, because Oracle fully supports spatial data types and queries, as well as R-tree index.

To utilize the geospatial knowledgebase for comparing two entities based on their geocoordinates, the system needs to assign the best spatial extent possible to a new incoming record or query. It also needs to support the efficient comparisons between two entities based on their assigned spatial extents. To achieve this, we developed two functions Geo-populate and Geo-compare. Consider populating the EntityBases. Geo-populate analyzes the phone number of a given new record to obtain its area code; it then queries the geospatial knowledgebase with the given area code to discover the spatial extents (a point or a region) for the record (the second step in Figure 7). Geo-compare then utilizes Oracle spatial APIs to compute how close is the new record to the records stored in the EntityBases based on their

[1] http://earth-info.nga.mil/gns/html/
[2] http://irantom.ru
[3] http://www.iranembassy.hu/province.htm

spatial extents (the third step in Figure 7). Figure 7 summarizes the framework for utilizing geospatial information for record linkage. In the follow subsections we consider these operations in more detail.

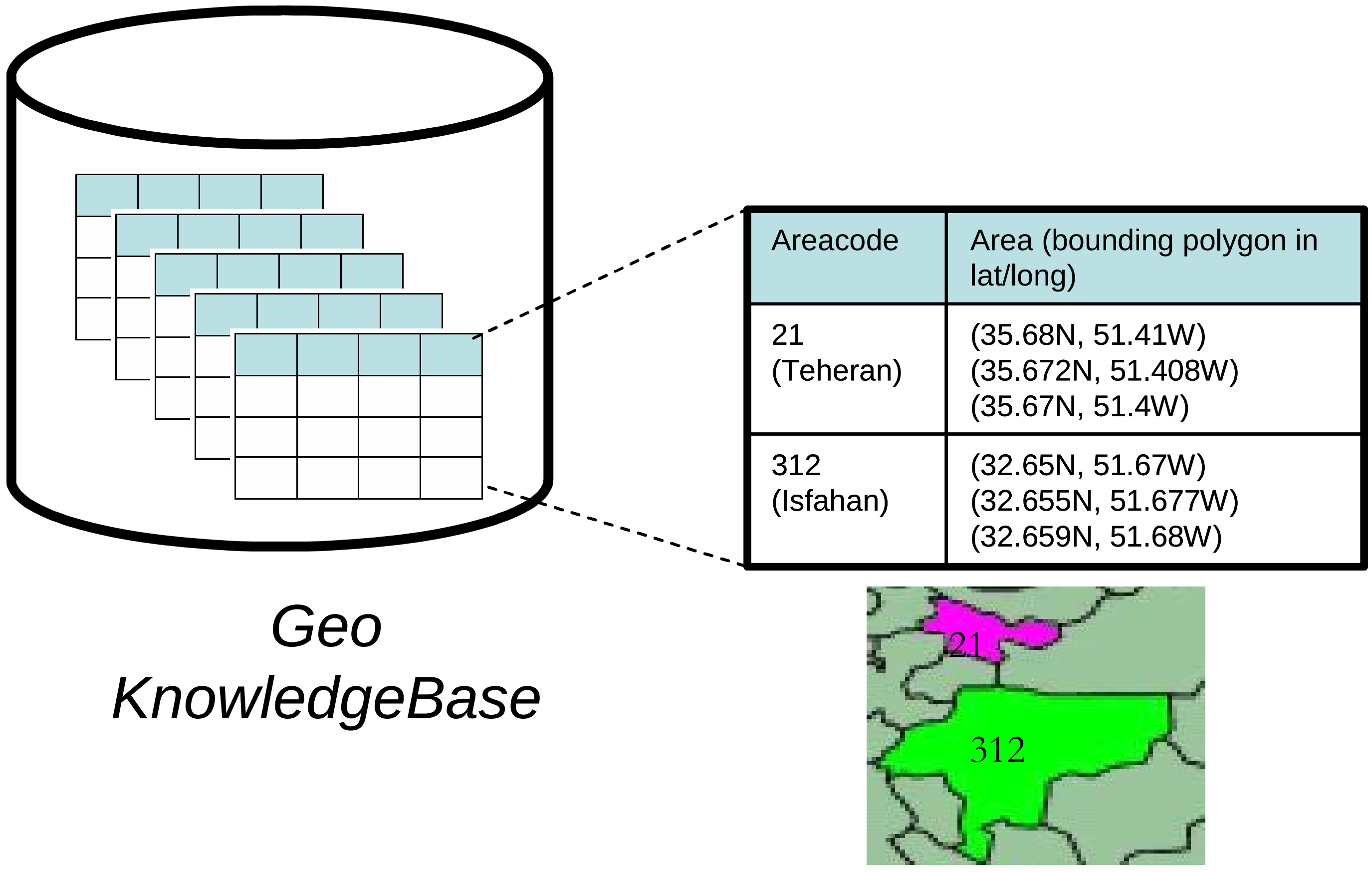


New record
RID | Name | Addr | Phone
2 Geo-populate
Utilized Spatial Data (e.g., NGA gazetteer, vector data, area code data)
1 Techniques to build geo knowledgebase
Geo knowledgebase
RID | Name | Addr | Phone | Spatial Extent
New record with spatial extent
3 Geo-compare
RID | Name | Addr | Phone | Spatial Extent
Existing EntityBase
Constructing Geo knowledgebase
Utilizing Geo knowledgebase for record linkage

Figure 7: Exploiting geospatial data for entity linkage

3.5.1 Construction of Geo Knowledgebase

A large number of geospatial data and related online sources are now available. The key questions are how to efficiently utilize and integrate these spatial data with entity information. To facilitate the use of various spatial data, we are building a comprehensive spatial database (called Geo Knowledgebase) that contains spatial knowledge covering the world. The database stores well-organized spatial knowledge relevant to the geo-locations of the textural geographic information (e.g., phone numbers or mailing addresses) about the world. For example, as shown in Figure 2, the geo knowledgebase links the phone area codes with the corresponding geo-coordinates. The system can then use this information to infer the location of a company based on its phone area code.



Areacode	Area (bounding polygon in lat/long)
21 (Teheran)	(35.68N, 51.41W) (35.672N, 51.408W) (35.67N, 51.4W)
312 (Isfahan)	(32.65N, 51.67W) (32.655N, 51.677W) (32.659N, 51.68W)

Figure 8: Example of data stored in *Geo KnowledgeBase*

To efficiently construct a scalable *geo knowledgebase,* we utilize two techniques:

1. Build thematic maps (e.g., area code maps or postal code maps) by relating gazetteer points to partitioned areas: The basic idea is to use available gazetteers databases to automatically produce thematic maps. Gazetteers databases (e.g., NGA gazetteers database) often store the named areas/organizations (e.g., schools) all over the world with associated geo-coordinates, while these areas/organizations are often associated mailing addresses and phone numbers. Therefore, using these geo-referenced points, we can reason about the areas (e.g., area code areas or postal code areas) with appropriate geo-coordinates and populate the *geo knowledgebase*.
2. Extract features from raster maps by identifying the area borders and labels: If the gazetteers are inaccessible, we can utilize raster thematic maps available online. However, the information contained in a raster map is often locked up in a raster image. We developed techniques to automatically detecting the area boundaries and labels from raster thematic maps and populate the *geo knowledgebase*.

3.5.2 Utilization of Geo Knowledge: *GeoPopulate*

Once we have built the *geo knowledgebase,* the system then uses the API *GeoPopulate* to link a record's textual geographic information (e.g., phone numbers) to spatial extents (e.g., geo-coordinates) by using the knowledge stored in *geo knowledgebase*. More precisely, *GeoPopulate* parses the records to find associated textual geographic data (e.g., area codes from the phone numbers). Subsequently, it lookups the *geo knowledgebase* to locate the corresponding geo-coordinates.

3.5.3 Utilization of Geo Knowledge: *GeoCompare*

Once the data records are populated into the *geo knowledgebase* with associated spatial extents, we can then use *GeoCompare* to compare any two records with spatial extents. For example, a user may be interested in the companies around the city, Tehran in Iran. *GeoCompare* can then use the spatial extents to support this sort of geospatial queries.

4 EntityBase – Significance of the Work

In this paper we have described our approach to building massive entity repositories. In particular, we have described the representation of the entities, the approach to linking new data into an EntityBase, a framework for querying the large amount of data potentially available on an entity, and a technique for exploiting the geospatial context to improve the linking of entities.

Previous research on record linkage has developed a foundation for statistically linking references across multiple databases, referred to variously as record linkage, consolidation or object identification (Fellegi et al. 1969; Winkler 1994; Goldberg et al. 1995; Tejada et al. 2001). While similar to traditional record linkage in the goal of identifying consolidating objects, the EntityBase design contains some key differences. In particular, the matching (linkage) process of EntityBases that we describe here is distinctly different than it is for traditional record linkage. In traditional record linkage, the goal is to identify the unique objects between two sources, where the objects are generally encapsulated as rows. In contrast, an EntityBase is composed of multiple sources, where each entity can have multiple attribute values (e.g., multiple names). Unlike record linkage, EntityBases keeps track of all object aliases and views the entire data collection as part of the entity.

A critical focus of the EntityBases work is on scale. We are attempting to build EntityBases of millions of entities, which can be queried and updated tens to hundreds of times per second. There is some past work on parallel record linkage (Christen et al. 2004) and blocking techniques (Baxter et al. 2003). However, these systems assume that the sources to be consolidated are tables in a relational database and do not address the issue of multi-valued attributes, which can have a serious performance impact. Furthermore, they do not consider the issues of entity merging and splitting, as we need to do in EntityBases.

Our data integration approach builds upon our previous work on the Prometheus mediator (Thakkar et al. 2005). Prometheus supports both the global-as-view and local-as-view approaches to data integration (Halevy 2001). For local-as-view integration, Prometheus follows the Inverse Rules algorithm (Duschka 1997) with additional optimizations (Thakkar et al. 2005). We use the mediator to assign common semantics to the data coming from the sources and map such data into the mediated schema of the EntityBase. Our mediation approach introduces two novel techniques. First, the mediator integration program is declaratively expressed in datalog including predicates that call the record linkage routines when a new source record is processed by the EntityBase. Second, the mediated schema is designed to accommodate the multivalued nature of attributes in the EntityBase.

There are many studies related to exploiting geospatial context to infer geospatial knowledge (Bruin 2000; Krieger et al. 2001; Sharifzadeh et al. 2003). For example, (Sharifzadeh et al. 2003) proposed a technique to construct thematic maps for non-natural features (e.g., zip code maps or area code maps). In general, we are not aware of previous work on exploiting these geospatial extents for matching entities.

One can build an EntityBase for just about any type of entity, including people, organizations, companies, terrorist groups, and so on. These knowledge bases of entities can then be used for a wide variety of applications. In this paper we described how an EntityBase can be used for associating and linking text documents with the actual entities that are mentioned in the documents. An EntityBase could also be used to process data in a database or to reason about the relationships between entities (such as finding all organizations that are located in the same region and are mentioned in the same document).

In future work, we plan to further develop the techniques and algorithms described in this paper. In particular, we plan to improve the accuracy of the linking process, extend the techniques for rapidly incorporating new sources of data, and exploit additional types of geospatial information. We also plan to refine the existing system to support the efficient construction and querying of EntityBases of millions of entities.

5 Smart Copy and Paste: Technical Description

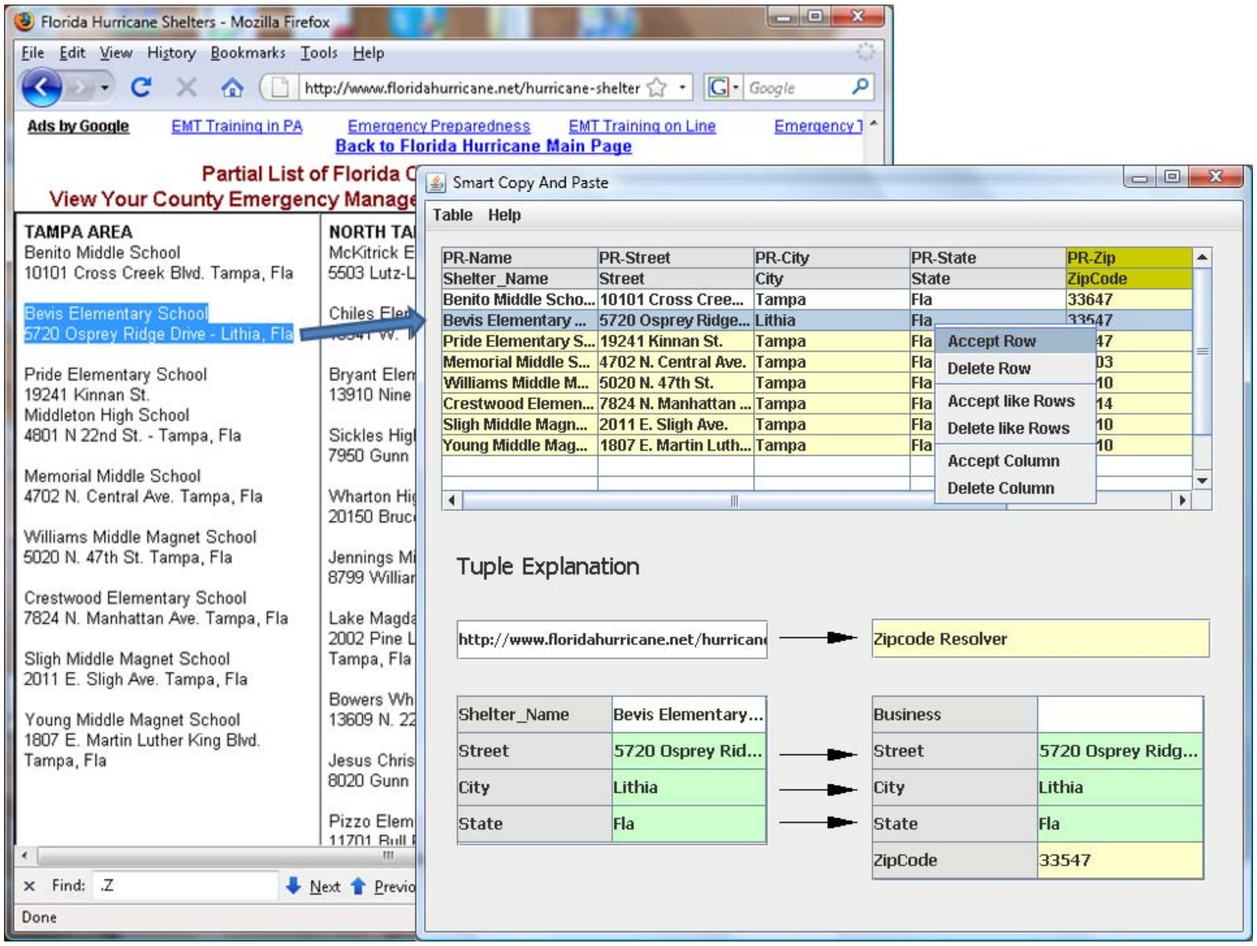
The data integration problem comes in many forms, and no single solution addresses all requirements. For instance, enterprisewide information integration typically involves gathering large volumes of data from tightly controlled sources, in order to perform OLAP queries or other analysis. Another common scenario is that business or scientific data is shared among a small number of parties, and portals or applications are developed to support a limited number of predetermined queries. Our focus in this project is on the class of data integration problems where a moderate number of Web and document sources (each with MB of data, but probably not GB) need to be integrated in a time-sensitive manner, possibly with small databases. In our target settings, a set of sources might be integrated on-demand to answer a specific query or class of queries; or they might be integrated in a way that rapidly evolves the mediated schema as new data is incorporated and new questions are posed.

Smart copy and paste is a form of programming by demonstration (Cypher et al., 1993, Lieberman, 2001, Tuchinda et al., 2008) (PBD): users demonstrate the actions to be performed to integrate data (copying data from source applications to the SCP workspace). The system learns to generalize their actions. Then the system immediately shows the effects of applying these generalizations, in the form of auto-complete suggestions, and solicits feedback on these suggestions. Through provenance information, the feedback can be "traced back" from data to the transformations (mappings) and extractors that are responsible for the data.

In the following section we describe the SCP user interface, introduce the basic architecture of a generic SCP system, and finally outline how our CopyCat implementation works. In subsequent sections we describe the operation of the main modules in our system.

5.1 User Interaction with SCP

To illustrate how a user interacts with CopyCat, we present a screenshot in Figure 9. Corresponding to the first steps of Example 1, the user has pasted into the table at the top of the window (the Workspace) several entries from the list of shelters. At this point, CopyCat has learned the datatypes and assigned attribute names for Street, City, and State; the user has manually entered the label for Shelter Name.



TAMPA AREA
Benito Middle School
10101 Cross Creek Blvd. Tampa, Fla

Bevis Elementary School
5720 Osprey Ridge Drive - Lithia, Fla

Pride Elementary School
19241 Kinnan St.
Middleton High School
4801 N 22nd St. - Tampa, Fla

Memorial Middle School
4702 N. Central Ave. Tampa, Fla

Williams Middle Magnet School
5020 N. 47th St. Tampa, Fla

Crestwood Elementary School
7824 N. Manhattan Ave. Tampa, Fla

Sligh Middle Magnet School
2011 E. Sligh Ave. Tampa, Fla

Young Middle Magnet School
1807 E. Martin Luther King Blvd.
Tampa, Fla

NORTH TA
McKitrick E
5503 Lutz-L

Chiles Elen
Bryant Elen
13910 Nine
Sickles Higl
7950 Gunn
Wharton Hi
20150 Bruc
Jennings M
8799 Willia
Lake Magd
2002 Pine L
Tampa, Fla
Bowers Wh
13609 N. 22
Jesus Chris
8020 Gunn
Pizzo Elem
11701 Bull

PR-Name	PR-Street	PR-City	PR-State	PR-Zip
Shelter_Name	Street	City	State	ZipCode
Benito Middle Scho...	10101 Cross Cree...	Tampa	Fla	33647
Bevis Elementary ...	5720 Osprey Ridge...	Lithia	Fla	33547
Pride Elementary S...	19241 Kinnan St.	Tampa	Fla	47
Memorial Middle S...	4702 N. Central Ave.	Tampa	Fla	03
Williams Middle M...	5020 N. 47th St.	Tampa	Fla	10
Crestwood Elemen...	7824 N. Manhattan ...	Tampa	Fla	14
Sligh Middle Magn...	2011 E. Sligh Ave.	Tampa	Fla	10
Young Middle Mag...	1807 E. Martin Luth...	Tampa	Fla	10

http://www.floridahurricane.net/hurrican → Zipcode Resolver

Shelter_Name	Bevis Elementary...
Street	5720 Osprey Rid...
City	Lithia
State	Fla

Business	
Street	5720 Osprey Ridg...
City	Lithia
State	Fla
ZipCode	33547

Figure 9: Screenshot of CopyCat as shelters are being added

In this example, CopyCat has existing knowledge of several data sources and Web services, including a Zipcode resolver that uses Google Maps to find zip codes using address information. It thus suggests a Zipcode attribute (rightmost attribute, highlighted in yellow) as the most promising join auto-completion, possibly from among several alternatives. Additionally, the system suggests row auto-completions (the rows after the highlighted cursor), by finding an information extraction pattern that returns the first two shelters plus additional ones. If we look closely at the example, we will see that the list of shelters in the Web document is quite irregular —in particular, Pride Elementary School has no city and state, and no separation from Middleton High School. The initial extractor learns patterns that give “mostly correct” information, but in this case the data is incomplete and the user will need to provide more examples.

In some cases, there may in fact be erroneous tuples in the suggestions. CopyCat emphasizes helping the user understand the presence of a suggested tuple or set of attributes, so he or she can provide feedback on what has been integrated. The Tuple Explanation pane (bottom of the figure) visualizes the provenance of the selected tuple in the table. Four attributes originate from the Web page of shelters (shown as a URL at the top of this pane, with the attributes below in a table). The Street, City, and State values are fed into the Zipcode Resolver (a dependent join; illustrated by the directed arrows to attributes in the rightmost table, and the green color scheme), which yields a Zip attribute. From the pop-up context menu, the user may explicitly accept or reject a single autocomplete row; accept or reject all rows similarly derived; and accept or reject the suggested (Zip) column. In all cases, the feedback operation is fed to the learning components of the SCP system, as discussed in the next section. CopyCat will revise its auto-complete suggestions accordingly.

The user is also allowed to ignore auto-complete suggestions and simply paste over them with new content. Such content may come from a different data source, forming a union (if a new row is pasted) or a join (if a new column is pasted). For a newly pasted column, the system attempts to determine possible sequences of joins and/or record linking operations by which data from the sources can meaningfully be combined.

5.2 Generic SCP Architecture

Figure 10 illustrates the architecture of an SCP system. Copy and paste operations — between source applications and the SCP workspace — are detected by application wrappers. Monitored operations, as well as context information like the document being displayed in the source application, are fed into three learner modules.

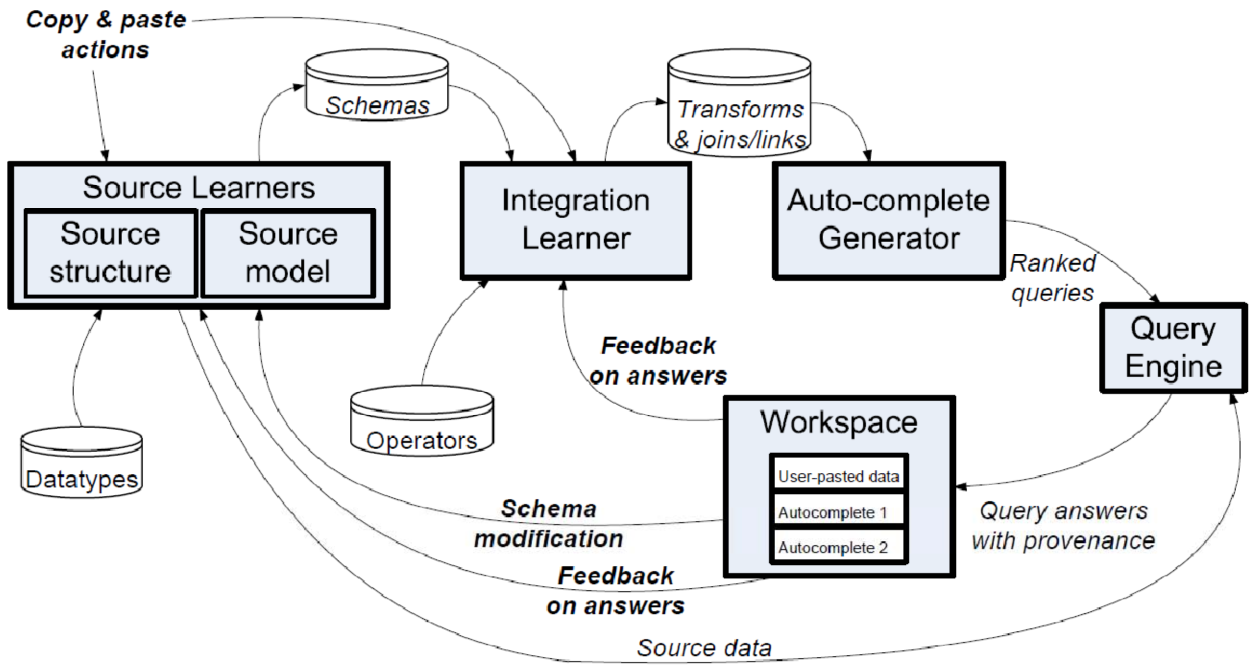


Figure 10: Architecture diagram for SCP system, with user interactions highlighted in boldface

Two learners focus on properties of individual sources: the context and structure learner learns extractors and any necessary input bindings required by the source, and the source model learner determines attributes and a schema for the source. The resulting source description gets added to a system catalog.

The integration learner determines the query and/or set of mappings that the user is creating by copy-and-paste. If the user performs a series of copy-paste-search-copy-paste operations, perhaps each intervening search represents a dependent join: attributes from the first source are looked up in the second source, and so on. In some other cases the user wants to perform a record linking or approximate join operation: here the SCP system can attempt to learn a record linking function from a set of examples — or, in some cases, use a function from a predefined library.

Once a few examples have been provided, the learners attempt to generalize from them, finding potential extractors and transformations. A ranked set of promising extractors and queries is produced by the auto-complete generator.

In turn these queries are run by the query engine to produce example answers, which are output to the user as extra rows and columns in the workspace. The user may provide feedback: promoting or demoting tuples; modifying the headings or data type specifiers for the columns; or adding or removing columns — all providing more detailed information to the learners in the system. Through data provenance (Buneman et al., 1997, Cui, 2001, Ives et al., 2008), feedback on tuples can be related back to the tuples' source queries. The learners adjust source scores, extraction patterns, and record linking or join conditions, in order to respond to the user feedback.

5.3 CopyCat: A Prototype SCP System

In the remainder of this paper, we describe the details of our initial CopyCat prototype, which builds upon the authors' prior experience with learning-based data integration tools. Our focus in this prototype is on the coupling between the clipboard, the workspace/user interface, and the learning systems.

Application wrappers. The initial CopyCat prototype supports monitoring of copy operations from a variety of common applications: Web browsers like Internet Explorer, and any HTML forms or pages they display; and Microsoft Office applications like Word and Excel.

Context and structure learner. CopyCat is given direct access to the underlying data being displayed in the browser or application window, as well as the data being copied. Our context and structure learner seeks to identify the origin of the copied data and to generalize the extraction operation(s) being performed, across the source data and — for multi-page sources — the source hierarchy.

Source model learner. This component detects the types of the data items being manipulated, and ultimately a source model that describes the function performed by the source. It uses this information to help find possible associations among data items it knows, or operations that can be performed to find related data: for instance, a join might be used across sources on social security number; a phone number might be looked up in a reverse directory to find a person; a record-linking function might match addresses.

Integration learner. Given a set of possible associations with different scores, the integration learner determines the most promising queries, which define auto-completions. It processes user feedback over these auto-completions to re-weight the scores of different associations, thus learning the user's preferred set of associations and integration queries.

Query engine. CopyCat employs the ORCHESTRA query answering system (Ives et al., 2008), which builds a layer over a relational DBMS to annotate every answer with data provenance. As described previously, provenance enables CopyCat to convert feedback on auto-complete data into feedback over the queries that created the data.

Workspace. The initial CopyCat interface is implemented in Java Swing, and is shown in Figure 9. While it provides both row- and column-auto-completions, its emphasis is on suggesting columns one at a time.

In the remainder of this report, we describe how the three learning components interoperate with the workspace. We refer the reader to (Talukdar et al., 2008) for an overview of how data provenance is recorded and maintained by the query processor.

5.4 Learning about Sources

Logically, each source has associated with it an extractor and a schema (possibly including input binding restrictions). The context and structure learner is primarily responsible for learning the extractor, and the source model learner is primarily responsible for learning the schema; however, the two modules are closely linked and share information.

5.4.1 Context and Structure Learner

The context and structure learner (CSL) analyzes data that is copied into the CopyCat workspace. By analyzing the data and the context from which it was copied, CSL can generalize the copy-and--paste operation so that a set of auto-complete suggestions can be provided to the user.

For a relatively structured source such as an Excel spreadsheet, the generalization process is normally quite simple. For example, after copying just two data items from a column in spreadsheet, it is clear that the user's selection should be generalized to include all the additional rows in that column. For semi-structured sources, such as Web sites, the hypothesis space is much larger. For instance, in the example in Figure 9, the user copies the first two shelters in the Tampa Area column of the Web page. However, it is not immediately clear whether the proper generalization is just the leftmost column, or the entire list on the page. After each copy and paste operation, CSL guesses a generalization, and the user can effectively provide feedback to the system either by accepting or rejecting the auto completed suggestions. If the user rejects the suggestions, the system will choose another hypothesis and revise the suggestions. If the user pastes another data item into a row (either replacing a current suggestion or augmenting the current suggestions) the system will select new hypothesis and make appropriate suggestions.

CSL uses a semi-supervised learning approach, which is based on our previous work on extracting data from the Web. This involves analyzing the structure of a website to identify its relational structure, following the approach described in (Gazen and Minton, 2006). This gives us a set of hypotheses. Then, given one or more examples selected by the user, the system attempts to find a mostgeneral hypothesis consistent with the example selected by the user. If this method cannot find a consistent hypothesis, the system falls back on a completely supervised approach based on more traditional wrapper induction techniques (Muslea et al., 2001).

The advantage of our approach is that we need not assume that the desktop is completely instrumented. We assume that a Copy- Cat wrapper provides the CSL system access to the source from which the data was selected (e.g., the Web site, spreadsheet, document, etc.), but we do not need to know exactly where the data was cut-and-pasted from.

5.4.2 Source Model Learner

In order to understand what task the user is performing and to better support the user, the system attempts to learn a model of each source that the user is manipulating. As described in the next section, this will help the system find relevant associations across sources and will also make it possible to find alternative sources that perform the same or similar tasks. The first step in learning the source model is to recognize the semantic types of each of the columns of data that the user is manipulating (Lerman et al., 2007). For example, if we recognize that a particular field is a social security number, then when we consider potential associations, we can consider joining this field with a field in another source that also contains a social security number. The system recognizes the semantic types of each field by learning a set of patterns from previous examples. The individual types can be further combined into complex types, such as recognizing a date or street address. The automatic composition of simple types into complex types allows the system to recognize combinations that may not have been seen previously.

In addition to learning the semantic types of the data that the user is manipulating, the system all atttempts to learn the task that is being performed by the various sources. For example, we want to be able to learn that a source can map an address into the corresponding latitude and longitude coordinates (i.e., a geocoder) or a source can determine the zipcode for a street address. This will allow the system to propose sources that can fill in gaps for a user (e.g., if the zipcode is missing for some of the fields) or even propose replacement sources if a source is down, too slow, or does not provide a complete set of results. The system can learn the function performed by a source by relating it to a set of known sources (Carman and Knoblock, 2007). The new source would be described in terms of a set of existing sources. The system can then compare the inputs and outputs of the new source to the description in terms of the existing sources by executing each one and comparing the similarity of the results. In order to learn a description of a new source, the existing sources must be able to cover the same functionality, but they can be composed in novel ways.

5.5 Learning to Integrate

The integration learner was developed by Zack Ives and his colleagues at the University of Pennsylvania. The integration learner attempts to determine what integration query is being constructed, based on knowledge of data sources and possible means of combining data across sources; it executes the most likely queries and presents their results as auto-completions. At its core, this learner maintains a source graph, in which nodes describe the schemas of data sources and what we generically term services. Services can be modeled as relations that take input parameters (i.e., to use the normal data integration terminology, they have input binding restrictions). Predefined services include recordlinking functions, address resolution, geocoding, and currency and unit conversion; we also model Web forms as services. Edges describe possible means of linking data from one source to another, e.g., by joining or by passing parameters to a dependent source like a Web service. Edges receive weights defining how relevant they are to the integration operation being performed; the weights are typically pre-initialized to a default value and then adjusted through learning.

5.6 Smart Copy and Paste: Significance and Open Challenges

Best-effort information integration is sometimes considered the “next frontier” for information management: the term of dataspaces (Franklin et al., 2005) was proposed as a name for this vein of work. Research in this area has included support for lightweight and community-driven extraction (Shen et al., 2008) and mapping (Mc- Cann et al., 2008, Jeffery et al., 2008); hints and trails (Salles et al., 2007); probabilistic schema mappings (Dong et al., 2007) and mediated schemas (Sarma et al., 2008); and various efforts to integrate keyword search with integration. An excellent tutorial appeared in VLDB 2008 (Halevy et al., 2008). Our focus is on developing an integrated creation and query system with provenance and feedback — abolishing the divide between design-time, runtime, and debugging stages. This goal of integrated processing in CopyCat required significant extensions of the ideas and techniques first developed as stand-alone components in the authors’ previous work: programming-by-demonstration techniques from the Karma (Tuchinda et al., 2008, Tuchinda et al., 2007) mashup construction system, learning from feedback in the Q (Talukdar et al., 2008) ranked query answering system, learning techniques for information extraction (Gazen and Minton, 2006, Muslea et al., 2001), and learning of source models (Carman and Knoblock, 2007, Lerman et al., 2007).

Our data integration framework incorporates a combination of programming by demonstration (Cypher et al., 1993, Lau, 2001, Lieberman, 2001) and query by example (QBE) (Zloof, 1977), and it is based on our earlier work on building mashups by example in Karma. In programming by demonstration, methods and procedures are induced from users' examples and interaction. This approach can be effective in various domains (Gibson et al., 2007, Lau et al., 2004, Sugiura and Koseki, 1998) where users understand and know how to do such tasks. In CopyCat users may not know how to formulate queries and only interact with the system through the data. The interaction is in a table similar to previous work on QBE. However, QBE requires users to manually select data sources, while CopyCat induces the sources to use by example and guides users to fill in only valid values.

The major innovation of CopyCat and the smart copy and paste model is its unified, lightweight interface for performing a range of information integration tasks — each of which has typically been addressed by separate tool (often with its own learning component). In a sense, smart copy and paste is to data integration what a spreadsheet is to the database: a dynamic, user-editable workspace where data can be rapidly added, visualized, and reorganized.

The core of CopyCat is a framework for plugging in information extractors, source description learners, and query learners; here, as described earlier in this paper, we were largely able to incorporate state-of-the-art components developed previously. Such components have been experimentally validated in isolation: for instance, the information extraction components are a major element of Fetch Technologies' business; query auto-completions (as implemented in the Karma system (Tuchinda et al., 2008)) saved approximately 75% of keystrokes compared to manual integration of data by copy and paste; and learning of correct queries based on user feedback over answers coverges very quickly (as little as one item of feedback for a single query, and feedback on 10 queries to learn rankings for an entire family of queries) in real domains, such as biology (Talukdar et al., 2008).

Not surprisingly, the primary challenges in developing Copy- Cat were related to the user interface and experience, and the integrated processing of feedback across multiple learners and modules. Wherever possible, our goal was to follow the spreadsheet metaphor: the user should be able to ignore autocomplete suggestions and continue to paste new data; or to directly modify attribute labels or even modify imported data — in addition to providing more direct feedback. Converting user actions to input to the learners is something we continue to refine: currently, if the query under construction contains no joins, we send feedback on data or metadata to the source learners; if the query contains joins then we send feedback to the integration learner.

We believe the current version of CopyCat is really just the first step in a series of developments on the smart copy and paste model. Our initial experiences suggest many avenues of future work specifically on the SCP model—in addition to all of the challenges posed by the individual learning components (which are already the subject of work by the database and machine learning communities). We briefly discuss what we believe are the most important directions of work on SCP.

Increased complexity and scale. The types of integration scenarios we target with SCP are generally "lightweight" tasks that may involve a limited number of tasks and relatively small sources. As we increase the number of source, there will be increasingly many possible queries and extractors. Open questions are how to present this to the user, such that it remains manageable and understandable; and how to ensure that there is sufficient information for the learner to make useful decisions.

Advanced interactions. To make the learning problem more tractable and to make interactions less complex, CopyCat makes certain default assumptions (e.g., that a set of sources should be joined using the conjunction of all possible predicates). In some cases an advanced user might want to remove some of these assumptions; and in general we must consider how to balance between simplifying user choice and "overcommitting" to certain types of queries. Moreover, it will ultimately be important to allow advanced users to "undo" or edit certain portions of what they have demonstrated to the system, and perhaps even to see how each demonstration changes the set of auto-completions.

Complex functions/transforms. Sometimes the user will want to do complex operations that are difficult to demonstrate: for instance, perform an aggregation or evaluate an arithmetic expression. It is important to explore approaches to searching for possible functions (Kache et al., 2008), and also potentially for allowing an advanced user to input such functions as in a spreadsheet.

Feedback interaction. As mentioned above, our user interface currently sends feedback to specific learners depending on the presence or absence of joins. We believe that ultimately there should be mechanisms for the integration learner to pass feedback to the source learners, and vice versa. To the best of our knowledge, little research has been done on enabling learners to cooperate.

Data cleaning. Our current implementation of CopyCat focuses on tasks relating to integrating data, but does not include data cleaning capabilities. The basic SCP model can fairly easily be extended to support a data cleaning mode, as illustrated by the Karma system (Tuchinda et al., 2008). However, this requires switching from "integration mode" to "cleaning mode" (so the extraction and integration learners do not try to generalize cleaning operations that should only apply to a single tuple). An open question is whether the distinction between these modes can be abolished.

Smart Copy and Paste is a new, unified model of information integration that removes the separation between design-time and run-time tasks and components—based monitoring copy operations from applications, suggesting generalizations and auto-completions through machine learning, and processing feedback. The smart copy and paste model is especially appropriate for lightweight integration tasks, where the integrator can immediately see the effects of each design decision, and can refine accordingly. We have identified a number of key research directions to be pursued. Our initial prototype, CopyCat, validates the basic framework and user interface, and provides a strong foundation upon which further work can be developed.

6 REFERENCES

Ambite, J. L., C. A. Knoblock, I. Muslea and A. Philpot (2001). "Compiling Source Descriptions for Efficient and Flexible Information Integration." Journal of Intelligent Information Systems **16**(2): 149--187.

Ambite, J. L., C. A. Knoblock, M. Muslea and S. Minton (2005). "Conditional Constraint Networks for Interleaved Planning and Information Gathering." IEEE Intelligent Systems **20**(2).

Bakshi, R., C. A. Knoblock and S. Thakkar (2004). Exploiting online sources to accurately geocode addresses. The 12th ACM International Symposium on Advances in Geographic Information Systems (ACMGIS'04), Washington, D.C.

Baxter, R., P. Christen and T. Churches (2003). A Comparison of fast blocking methods for record linkage. ACM SIGKDD'03 Workshop on Data Cleaning, Record Linkage, and Object Consolidation, Washington, DC.

Bilenko, M. and R. J. Mooney (2003). Adaptive Duplicate Detection Using Learnable String Similarity Measures. The 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2003).

Bruin, S. D. (2000). "Predicting the areal extent of land-cover types using classified imagery and geostatistics." Remote Sensing of Environment **74**(3): 387–396.

Buneman, P., Davidson, S. B., Fernandez, M. F., and Suciu, D. (1997). Adding structure to unstructured data. In ICDT, volume 1186.

Carman, M. and Knoblock, C. A. (2007). Learning semantic definitions of online information sources. Journal of Artificial Intelligence Research (JAIR), 30.

Christen, P., T. Churches and M. Hegland (2004). A Parallel Open Source Data Linkage System. The 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'04), Sydney, Australia.

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., and Singer, Y. (2006). Online passive-aggressive algorithms. Journal of Machine Learning Research, 7:551–585.

Cui, Y. (2001). Lineage Tracing in Data Warehouses. PhD thesis, Stanford University.

Cypher, A., Halbert, D. C., Kurlander, D., Lieberman, H., Maulsby, D., Myers, B. A., and Turransky, A., editors (1993). Watch What I Do: Programming by Demonstration. The MIT Press. Available from http://acypher.com/wwid/.

Dong, X. L., Halevy, A. Y., and Yu, C. (2007). Data integration with uncertainty. In VLDB.

Duschka, O. M. (1997). Query Planning and Optimization in Information Integration. Department of Computer Science, Stanford University.

Fellegi, I. P. and A. B. Sunter (1969). "A theory for record-linkage." Journal of the American Statistical Association **64**: 1183-1210.

Franklin, M., Halevy, A., and Maier, D. (2005). From databases to dataspaces: a new abstraction for information management. SIGMOD Rec., 34(4).

Gazen, B. and Minton, S. (2006). Overview of autofeed: An unsupervised learning system for generating webfeeds. In AAAI.

Gibson, A., Gamble, M., Wolstencroft, K., Oinn, T., and Goble, C. (2007). The data playground: An intuitive workflow specification environment. In E-SCIENCE.

Halevy, A. Y. (2001). "Answering queries using views: A survey." The VLDB Journal **10**(4): 270-294.

Halevy, A. Y., Ives, Z. G., Suciu, D., and Tatarinov, I. (2003). Schema mediation in peer data management systems. In ICDE.

Halevy, A. Y., Maier, D., and Franklin, M. J. (2008). Dataspaces: The tutorial. In VLDB.

Ives, Z. G., Green, T. J., Karvounarakis, G., Taylor, N. E., Tannen, V., Talukdar, P. P., Jacob, M., and Pereira, F. (2008). The ORCHESTRA collaborative data sharing system. SIGMOD Rec.

Jeffery, S. R., Franklin, M. J., and Halevy, A. Y. (2008). Pay-as-you-go user feedback for dataspace systems. In SIGMOD.

Kache, H., Saillet, Y., and Roth, M. (2008). Transformation rule discovery through data mining. In International Workshop on New Trends in Information Integration.

Krieger, N., P. Waterman, K. Lemieux, S. Zierler and J. Hogan (2001). "On the wrong side of the tracts? Evaluating the accuracy of geocoding in public health research." American journal of public health **91**(7): 1114-1116.

Lau, T. (2001). Programming by Demonstration: a Machine Learning Approach. PhD thesis, University of Washington.

Lau, T., Bergman, L., Castelli, V., and Oblinger, D. (2004). Sheepdog: learning procedures for technical support. In IUI.

Lerman, K., Plangprasopchok, A., and Knoblock, C. A. (2007). Semantic labeling of online information sources. International Journal on Semantic Web and Information Systems, 3(3).

Lieberman, H., editor (2001). Your Wish is My Command: Programming by Example. Morgan Kaufman.

McCann, R., Shen, W., and Doan, A. (2008). Matching schemas in online communities: A web 2.0 approach. In ICDE.

Michelson, M. and C. A. Knoblock (2006). Learning Blocking Schemes for Record Linkage. The 21st National Conference on Artificial Intelligence (AAAI-06), Boston, MA.

Minton, S. N., C. Nanjo, C. A. Knoblock, M. Michalowski and M. Michelson (2005). A Heterogeneous Field Matching Method for Record Linkage. The 5th IEEE International Conference on Data Mining (ICDM-05), Houston, TX.

Muslea, I., Minton, S., and Knoblock, C. A. (2001). Hierarchical wrapper induction for semistructured information sources. Autonomous Agents and Multi-Agent Systems, 4(1/2).

Rahm, E. and Bernstein, P. A. (2001). A survey of approaches to automatic schema matching. VLDB J., 10(4).

Salles, M. A. V., Dittrich, J.-P., Karakashian, S. K., Girard, O. R., and Blunschi, L. (2007). iTrails: pay-as-you-go information integration in dataspaces. In VLDB.

Sarma, A. D., Dong, X., and Halevy, A. (2008). Bootstrapping pay-as-you-go data integration systems. In SIGMOD, New York, NY, USA.

Sharifzadeh, M., C. Shahabi and C. A. Knoblock (2003). Learning Approximate Thematic Maps from Labeled Geospatial Data. the International Workshop on Next Generation Geospatial Information, Cambridge (Boston), Massachusetts, USA.

Shen, W., DeRose, P., McCann, R., Doan, A., and Ramakrishnan, R. (2008). Toward best-effort information extraction. In SIGMOD, New York, NY, USA.

Sugiura, A. and Koseki, Y. (1998). Internet scrapbook: automating web browsing tasks by demonstration. In UIST.

Talukdar, P. P., Jacob, M., Mehmood, M. S., Crammer, K., Ives, Z. G., Pereira, F., and Guha, S. (2008). Learning to create data-integrating queries. In VLDB.

Tejada, S., C. A. Knoblock and S. Minton (2001). "Learning Object Identification Rules for Information Integration." Information Systems **26**(8).

Thakkar, S., J. L. Ambite and C. A. Knoblock (2005). "Composing, Optimizing, and Executing Bioinformatics Web Services." VLDB Journal, Special Issue on Data Management, Analysis and Mining for Life Sciences **14**(3): 330-353.

Tuchinda, R., Szekely, P., and Knoblock, C. A. (2007). Building data integration queries by demonstration. In IUI.

Tuchinda, R., Szekely, P., and Knoblock, C. A. (2008). Building mashups by example. In IUI.

Winkler, W. E. (1994). Advanced Methods for Record Linkage. Proceedings of the Section of Survey Research Methods of the American Statistical Association**:** 467--472.

Zloof, M. M. (1977). Query by example: A data base language. IBM Systems Journal, 16(4).